# LU02 Auftrag: Landingpage Alarado – Fonts, Variablen und Responsive

**Ziel:** Sie ergänzen im bestehenden LU01-Projekt die **TODO in LU02**:

1. Google Font einbinden
2. CSS-Variablen in `:root` definieren und verwenden
3. Typografie, Navigation und Buttons gemäss Design stylen
4. Responsive Anpassungen mit einer Media Query umsetzen

### Voraussetzungen

- Sie arbeiten im bereitgestellten `index.html` Template mit den TODO-Kommentaren.

Falls Sie in der letzten Unterrichtseinheit nicht weit gekommen sind, laden Sie sich dieses HTML-File herunter und verbessern Sie Ihren Code nach dieser Vorlage.

Code nach LU01

### Arbeitsweise

- Arbeiten Sie von oben nach unten im Template.
- Nach jedem Schritt: speichern → Browser aktualisieren → Ergebnis prüfen.
- Wenn etwas nicht funktioniert: DevTools öffnen → Console und Elements prüfen.

**Hilfreiche Links**

- Google Fonts: fonts.google.com
- MDN: CSS Custom Properties: MDN CSS Variables
- MDN: Media Queries: MDN Media Queries
- DevTools Responsive Mode: Chrome DevTools Device Mode

---

## Schritt 1 Google Font Poppins einbinden

### Aufgabe

- Öffnen Sie Google Fonts und suchen Sie **Poppins**.
- Wählen Sie passende Schriftschnitte aus, die Sie im Design benötigen (typisch: 400, 500, 600, 700).
- Kopieren Sie den vorgeschlagenen `<link>` Code und fügen Sie ihn im `<head>` an der Stelle **TODO in LU02** ein.

Optional können Sie zusätzlich `preconnect` Links verwenden, damit der Browser schneller eine Verbindung aufbaut. [1)]

### Checkpoint

- Seite neu laden: keine Fehler in der Console.
- Im Elements-Tab sehen Sie im `<head>` die Font-Links.

---

## Schritt 2 CSS-Variablen in :root definieren

### Aufgabe

- Füllen Sie den Block `:root` im CSS.
- Definieren Sie Variablen für:
  1. Farben für Hintergrund und Text
  2. Aktive Linkfarbe
  3. Schriftgrössen und Schriftgewichte für Navigation, h1, h3, Buttons
- Werte nehmen Sie aus dem Figma-Design.

**Beispielstruktur für :root**

```
:root {
  /* Farben */
  --color-bg-light: ...;
  --color-text-light: ...;
  --color-accent: ...;

  /* Typografie */
  --size-h1-desktop: ...;
  --size-h1-mobile: ...;
  --weight-h1: ...;

  --size-h3: ...;
  --weight-h3: ...;
```

```
  --size-nav: ...;
  --weight-nav: ...;

  --size-button: ...;
  --weight-button: ...;
}
```

**Regel:** Ab jetzt verwenden Sie im CSS möglichst oft `var(–…)` statt einzelne Zahlen zu kopieren.

### Checkpoint

- Sie haben mindestens: Hintergrund, Text, Accent-Farbe sowie Variablen für h1/h3/nav/button.

---

## Schritt 3 Grundstyles im body setzen

### Aufgabe

- Setzen Sie im `body`:
  1. `font-family` auf Poppins mit sinnvoller Fallback-Schrift
  2. `background-color` und `color` über Ihre Variablen
  3. `line-height` für gute Lesbarkeit

**Beispiel welche Eigenschaften erwartet werden**

```
body {
  font-family: ...;
  background-color: var(...);
  color: var(...);
  line-height: ...;
}
```

### Checkpoint

- Text wirkt sichtbar in der neuen Schrift.
- Hintergrund und Textfarbe stimmen mit dem Design überein.

---

# Schritt 4 Navigation stylen

### Aufgabe

- Stylen Sie die Navigation so, dass:
  1. die `li` eine definierte Schriftgrösse und ein Gewicht erhalten (über Variablen)
  2. Links standardmässig keine Unterstreichung haben
  3. der aktive Link als aktiv erkennbar ist (Farbe und Unterstreichung gemäss Design)
- Achten Sie darauf, dass nur der aktive Link speziell gestylt ist.

Für „alle Links ausser active“ eignet sich ein Selektor wie `a:not(.active)`.[2)]

### Checkpoint

- Nur „About us“ ist aktiv gestylt.
- Die anderen Links sehen wie normale Navigation aus.

---

# Schritt 5 Typografie für h3 und h1

### Aufgabe

- Setzen Sie bei h3:
  1. Schriftgrösse und Gewicht über Variablen
  2. Grossschreibung mit `text-transform: uppercase`
  3. passende Abstände (margin)
- Setzen Sie bei h1:
  1. Schriftgrösse und Gewicht über Variablen
  2. passende Abstände gemäss Design

**Beispiel welche Eigenschaften erwartet werden**

```
h3 {
  font-size: var(...);
  font-weight: var(...);
  text-transform: uppercase;
  margin-bottom: ...;
}

h1 {
  font-size: var(...);
  font-weight: var(...);
```

```
  margin-bottom: ...;
}
```

### Checkpoint

- h3 wirkt wie im Design (klein, fett, gross geschrieben).
- h1 wirkt wie ein klarer Haupttitel (gross, stark gewichtet).

---

## Schritt 6 Buttons stylen

### Aufgabe

- Definieren Sie Basis-Styles für alle `button`:
  1. Schriftgrösse und Gewicht über Variablen
  2. Cursor-Verhalten
- Stylen Sie anschliessend die Varianten:
  1. `.highlight` als primärer Button (Padding, Hintergrundfarbe, Radius, Textfarbe)
  2. `.active` als sekundärer Button (transparent, keine Border)

**Beispiel welche Bereiche Sie umsetzen sollen**

```
button { ... }
button:hover { ... }

button.highlight { ... }
button.active { ... }
```

### Checkpoint

- „Get Started“ wirkt wie ein Hauptbutton.
- „Get live demo“ wirkt wie ein zurückhaltender Zweitbutton.

---

## Schritt 7 Responsive Regeln mit Media Query

### Aufgabe

- Ergänzen Sie im Template den `@media` Block so, dass das Mobile-Layout funktioniert:
  1. Hero wird einspaltig
  2. Bild kommt oben, Text darunter
  3. Innenabstände werden kleiner
  4. Navigation-Liste und Switch verschwinden
  5. Hamburger-Icon wird sichtbar

6. h1 wird kleiner

**Gerüst für den Media Query Block**

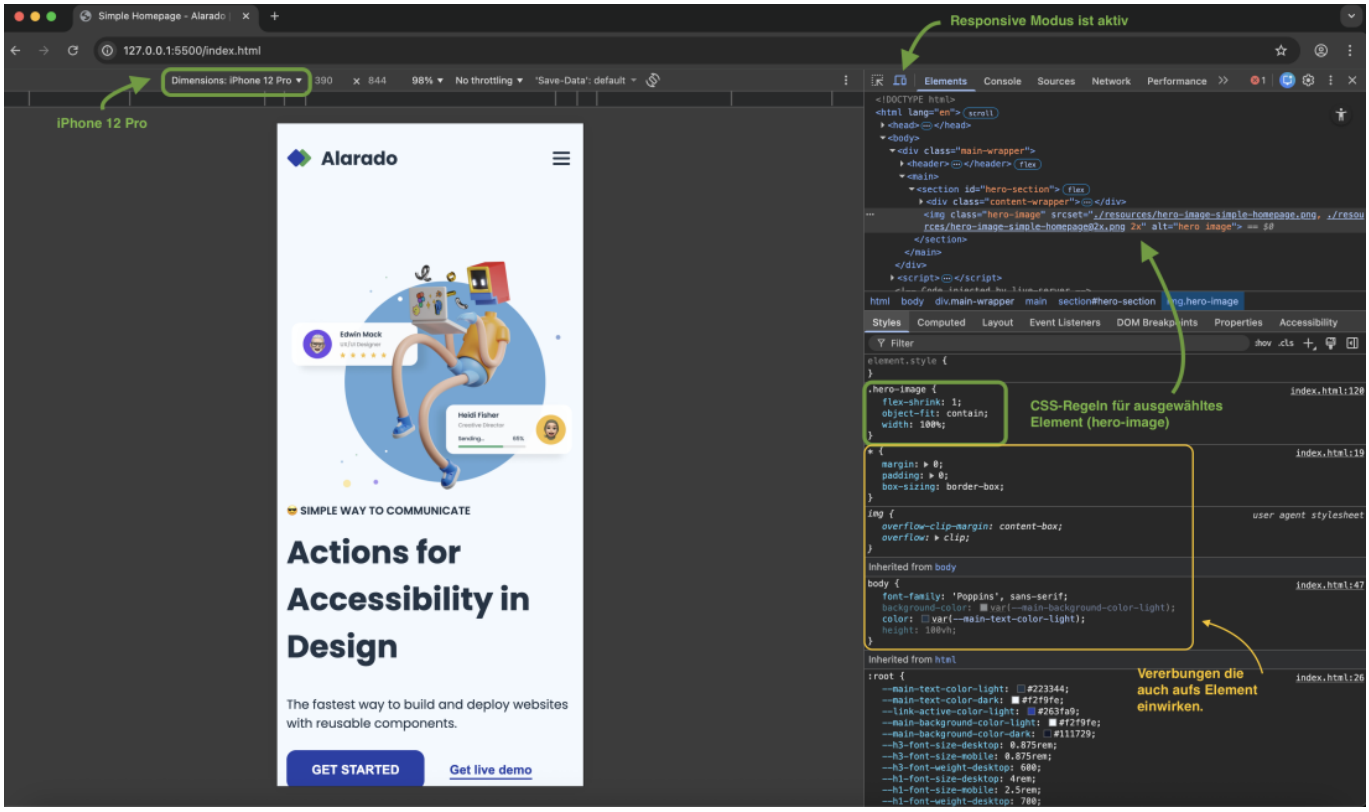
```
@media screen and (width < 1024px) {
  /* Hero Layout anpassen */
  /* Content-Wrapper Breite anpassen */
  /* Container Padding anpassen */
  /* Typografie Mobile anpassen */
  /* Navigation/Switch ausblenden, Hamburger einblenden */
}
```

**Kaskade merken:** Wenn ein Selektor im `@media` Block gleich ist wie oben, überschreibt die Regel im `@media` Block die vorherige Regel, sobald die Bedingung stimmt.

### Checkpoint

- Über 1024px: 2-Spalten Layout.
- Unter 1024px: 1-Spalten Layout mit Bild oben.
- Hamburger sichtbar, Menu und Switch versteckt.

## Schritt 8 Responsive testen mit DevTools




### Aufgabe

- Öffnen Sie DevTools und aktivieren Sie den Device Toolbar Modus.
- Testen Sie mindestens:
  1. Breite knapp über 1024px
  2. Breite knapp unter 1024px
  3. ein Smartphone-Preset (z.B. iPhone)

### Achten Sie besonders auf Folgendes:

- Kein horizontales Scrollen.
- Keine überlappenden Elemente.
- Text bleibt gut lesbar.

# Schritt 9 Abgabe und Commit

### Abgabe Ende LU02

- Commit-Message: `LU02: fonts variables responsive`
- Zeigen Sie der Lehrperson kurz:
  1. Poppins aktiv
  2. Variablen in `:root` vorhanden und verwendet
  3. Navigation und Buttons gestylt
  4. Mobile Layout funktioniert

1)

`preconnect` ist ein Performance-Hinweis: Der Browser startet die Verbindung früher.

2)

Der Selektor trifft alle Links, die nicht die Klasse active haben.